Pertemuan 5

Pencarian Heuristik dan Algoritma Informasi

A. Konsep Heuristik dan Fungsi Evaluasi

1. Pengantar

Pada pertemuan sebelumnya, kita telah membahas pencarian buta (*uninformed search*) seperti BFS dan DFS, yang bekerja tanpa informasi tambahan tentang posisi *goal*. Namun, dalam kasus nyata — seperti navigasi robot, rute GPS, atau pencarian solusi pada permainan catur — diperlukan pendekatan yang lebih cerdas untuk mempercepat pencarian. Pencarian heuristik (heuristic search) disebut juga *informed search*, karena menggunakan informasi tambahan (heuristik) untuk memperkirakan seberapa dekat suatu state dengan goal. Dengan heuristik, sistem AI dapat memilih jalur yang lebih menjanjikan, bukan sekadar mencoba semua kemungkinan seperti BFS/DFS.

2. Pengertian Heuristik

Kata *heuristic* berasal dari bahasa Yunani *heuriskein*, yang berarti *menemukan* atau *menemukan jalan*. Heuristik adalah *fungsi estimasi* yang digunakan untuk menilai seberapa baik suatu keadaan (*state*) dalam mendekati tujuan (*goal*). Heuristik tidak selalu benar atau akurat, tetapi memberikan petunjuk (*guidance*) agar proses pencarian menjadi lebih efisien.

Contoh Kasus

Bayangkan Anda ingin menemukan rute tercepat dari rumah ke kampus:

- BFS: akan menelusuri semua jalan yang mungkin, tanpa tahu mana yang cepat.
- Heuristic Search: akan memperkirakan jarak terpendek ke kampus dari setiap titik dan memilih jalur yang *lebih menjanjikan*.

Jadi, heuristik membantu sistem berpikir “cerdas”, bukan “brute-force”.

3. Perbedaan Pencarian Buta dan Pencarian Heuristik

Aspek	Uninformed Search	Informed / Heuristic Search
Informasi tentang goal	Tidak memiliki informasi tambahan	Menggunakan estimasi jarak ke goal
Proses	Mencoba semua kemungkinan	Memilih arah paling menjanjikan
Contoh algoritma	BFS, DFS	Best-First Search, A*, Hill Climbing
Efisiensi	Lambat, karena eksplorasi luas	Cepat, karena fokus pada jalur potensial
Optimalitas	Kadang tidak optimal	Dapat mendekati optimal (tergantung heuristik)

4. Fungsi Heuristik (*Heuristic Function*)

Fungsi heuristik adalah fungsi yang memberikan nilai estimasi (h(n)) terhadap biaya terpendek dari node saat ini ke goal.

h(n) = estimasi biaya dari *node n* menuju *goal*

Makna Nilai h(n)

- h(n) kecil → node tersebut dekat dengan goal
- h(n) besar → node tersebut jauh dari goal

Contoh

Masalah pencarian rute kota

Kota	Estimasi jarak (heuristik) ke "Kolaka"
Kendari	180 km
Lasusua	120 km
Pomalaa	30 km
Kolaka	0 km

Fungsi heuristik h(n) memberi "petunjuk jarak" agar algoritma lebih cepat menuju *goal* (Kolaka).

5. Fungsi Evaluasi (*Evaluation Function*)

Fungsi evaluasi digunakan untuk memilih node terbaik yang akan diekspansi (dilanjutkan penelusurannya) berdasarkan gabungan antara biaya aktual dan estimasi heuristik.

Rumus umum:

$$f(n) = g(n) + h(n)$$

Keterangan:

- f(n) = total biaya evaluasi
- g(n) = biaya aktual dari *start* ke *n*
- h(n) = estimasi biaya dari *n* ke *goal*

Interpretasi:

Komponen	Arti
g(n)	Sudah pasti (biaya nyata dari awal ke titik saat ini)
h(n)	Perkiraan (biaya sisa menuju tujuan)
f(n)	Estimasi total biaya dari awal sampai tujuan melalui node tersebut

Semakin kecil nilai f(n), semakin "menjanjikan" node tersebut untuk dijelajahi.

Contoh Perhitungan

Semua memiliki nilai f(n) sama, maka sistem dapat memilih salah satu jalur berdasarkan urutan prioritas.

6. Sifat Heuristik yang Baik

Agar algoritma pencarian dapat bekerja dengan efisien dan tetap akurat, fungsi heuristik harus memiliki beberapa sifat berikut:

Sifat	Penjelasan
Admissible (Dapat diterima)	h(n) tidak melebihi biaya sebenarnya menuju goal (optimistik).
Consistent (Konsisten)	Untuk setiap transisi n → n', nilai heuristik memenuhi:

Contoh Sifat Admissible:

Jika jarak sebenarnya antara dua kota adalah 100 km, maka h(n) = 80 km (optimistik, lebih kecil dari aktual)

→ masih dapat diterima.

Jika h(n) = 120 km (melebihi aktual)

→ tidak admissible (terlalu pesimistis, bisa melewatkan solusi terbaik).

7. Hubungan antara f(n), g(n), dan h(n)

Diagram hubungan:

```
Start -----(g(n))----> Current Node -----(h(n))----> Goal
          <------ f(n) = g(n) + h(n) ------>
```

Visualisasi:

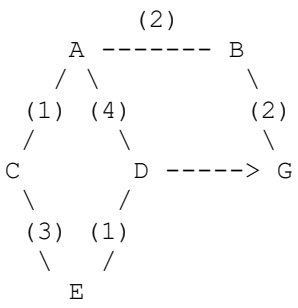
- g(n): usaha yang sudah dilakukan
- h(n): perkiraan usaha yang tersisa
- f(n): total estimasi perjalanan

8. Contoh Aplikasi Heuristik dalam AI

Bidang	Penerapan Heuristik
Navigasi / GPS	Menggunakan jarak Euclidean (jarak garis lurus) sebagai h(n)
Game AI (misalnya Catur)	Estimasi skor posisi papan untuk memilih langkah terbaik
Robotika	Mengukur jarak antara posisi robot dan target
Natural Language Processing	Menilai kecocokan antara kata yang diketik dengan kamus (spell correction)
Machine Vision	Menilai kedekatan pola visual terhadap pola target

9. Ilustrasi Visual (Contoh Rute Kota)

Misal AI ingin mencari rute dari A → G



Node	g(n)	h(n) (jarak ke G)	f(n)
A	0	6	6
B	2	3	5
C	1	4	5
D	4	2	6
G	6	0	6

Node dengan f(n) terkecil (5) menjadi prioritas untuk diekspansi terlebih dahulu. Inilah prinsip dasar Best-First Search dan A* yang akan dibahas di bagian berikutnya.

B. Best-First Search, A* Algorithm

1. Pengantar

Setelah memahami konsep heuristik (h(n)) dan fungsi evaluasi (f(n)), langkah berikutnya adalah mempelajari dua algoritma utama dalam pencarian heuristik, yaitu:

1. Best-First Search (BFS heuristik)
2. A* (A-Star) Algorithm

Keduanya menggunakan nilai heuristik untuk memandu proses pencarian agar lebih cepat dan lebih efisien dibandingkan pencarian buta (BFS/DFS).

Tujuan utama algoritma heuristik:
Menemukan solusi optimal dengan eksplorasi minimal.

2. Konsep Best-First Search (BFS Heuristik)

Best-First Search adalah algoritma pencarian yang selalu memilih node yang paling menjanjikan, berdasarkan nilai heuristik terkecil h(n).

"Pilih node yang *tampak paling dekat* dengan goal berdasarkan perkiraan heuristik."

Prinsip Kerja

1. Mulai dari node awal (start).

2. Hitung h(n) untuk semua node tetangga.
3. Pilih node dengan h(n) terkecil (paling dekat ke goal).
4. Ulangi hingga goal ditemukan atau semua node dievaluasi.

Fungsi Evaluasi yang Digunakan

$$f(n) = h(n)$$

Artinya, hanya mengandalkan estimasi heuristik, tanpa mempertimbangkan biaya aktual (*g(n)*).

Contoh Kasus

Mencari rute dari A → G menggunakan data heuristik berikut:

Node	Tetangga	h(n) ke Goal
A	B, C	10
B	D, E	6
C	F	5
D	G	3
E	G	1
F	G	2
G	–	0

Langkah Pencarian Best-First Search:

1. Start di A → h(A) = 10
   → tetangga: B (6), C (5)
   Pilih C (karena 5 lebih kecil dari 6)
2. Dari C → tetangga: F (2)
   Pilih F
3. Dari F → tetangga: G (0)
   Goal ditemukan

Jalur Solusi: A → C → F → G

Kelebihan

- Sangat cepat pada ruang pencarian besar.
- Hanya memeriksa node "yang berpotensi".
- Cocok untuk sistem navigasi dan pencarian rute cepat.

Kelemahan

- Tidak mempertimbangkan biaya perjalanan aktual (*g(n)*).
- Bisa melewati jalur yang sebenarnya lebih murah.
- Tidak selalu menghasilkan solusi optimal.

3. Konsep A* (A-Star) Algorithm

A* adalah algoritma pencarian heuristik yang menggabungkan kelebihan Best-First Search dan Uniform Cost Search.

Menggunakan dua komponen:

- Biaya nyata dari awal → node saat ini (g(n))
- Estimasi biaya dari node saat ini → goal (h(n))

Sehingga fungsi evaluasinya:

$$f(n) = g(n) + h(n)$$

Makna:

Komponen	Makna
g(n)	Biaya aktual dari titik awal ke node saat ini
h(n)	Estimasi biaya dari node saat ini ke tujuan
f(n)	Estimasi total biaya jalur dari awal ke tujuan melalui node n

Node dengan nilai f(n) terkecil akan diekspansi lebih dahulu.

Langkah-langkah A* Algorithm

1. Mulai dari node awal (start node).
2. Hitung nilai f(n) = g(n) + h(n) untuk semua tetangga.
3. Pilih node dengan f(n) terkecil untuk dijelajahi berikutnya.
4. Perbarui nilai g(n) dan f(n) saat jalur baru ditemukan.
5. Ulangi hingga goal state tercapai.

4. Contoh Penerapan A*

Kasus

Cari jalur terpendek dari A ke G

Node	Tetangga (biaya g)	h(n) ke Goal
A	B(2), C(3)	6
B	D(4), E(3)	4
C	F(5)	4
D	G(2)	2
E	G(3)	1
F	G(5)	2
G	–	0

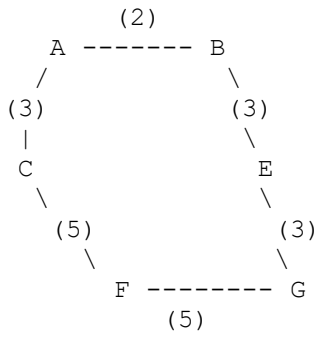
Langkah Perhitungan A* (Step-by-Step)

Jalur	g(n)	h(n)	f(n)=g+h	Catatan
A	0	6	6	Start
A → B	2	4	6	sama dengan A
A → C	3	4	7	lebih tinggi
A → B → D	6	2	8	
A → B → E	5	1	**6**	
A → B → E → G	8	0	8 (Goal)	☑

Jalur solusi terbaik:

A → B → E → G dengan total biaya 8

Penjelasan Visual



- A* mengevaluasi semua kemungkinan jalur.
- Jalur dengan nilai f(n) paling kecil diekspansi terlebih dahulu.

- Hasil akhirnya adalah rute biaya total minimum.

5. Sifat A* Algorithm

Sifat	Penjelasan
Optimal	Menemukan solusi terpendek jika heuristiknya *admissible* (tidak melebihi biaya sebenarnya).
Complete	Akan menemukan solusi jika solusi ada.
Efisien	Menggabungkan kekuatan pencarian luas (BFS) dan mendalam (Best-First).
Adaptif	Dapat digunakan pada graf berbobot dan tidak berbobot.

Contoh Heuristik Admissible untuk A*

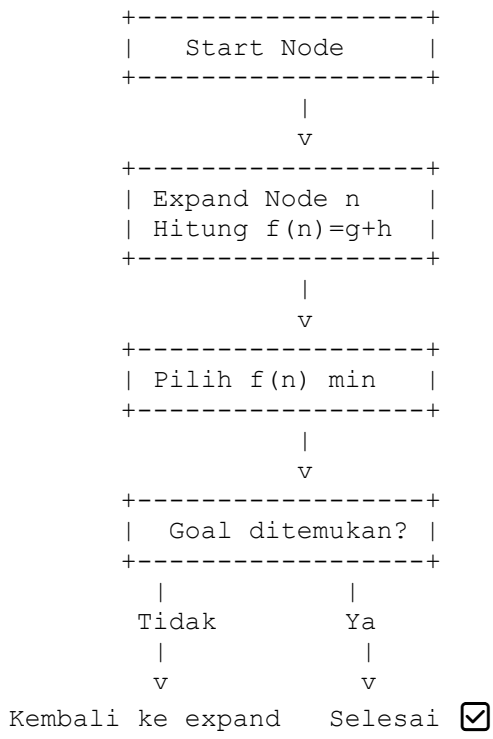
Jika jarak garis lurus antar kota ≤ jarak jalan sebenarnya, maka fungsi h(n) dapat diterima (*admissible heuristic*). Misal: jarak garis lurus Kendari–Kolaka (120 km) sedangkan jarak jalan sebenarnya 150 km

→ h(n) = 120 < 150 → admissible ☑

6. Perbandingan Best-First Search dan A*

Aspek	Best-First Search	A*
Fungsi evaluasi	f(n) = h(n)	f(n) = g(n) + h(n)
Fokus utama	Estimasi jarak ke goal	Total estimasi biaya
Pertimbangan biaya aktual	✗ Tidak dipertimbangkan	☑ Dipertimbangkan
Optimalitas	Tidak dijamin	Dijamin (jika heuristik admissible)
Efisiensi memori	Lebih ringan	Lebih berat
Contoh aplikasi	Navigasi cepat, game sederhana	Rute GPS, robotika, pathfinding kompleks

7. Visualisasi Alur Kerja A*



8. Kelebihan dan Kelemahan

Kelebihan	Kelemahan
Menemukan solusi optimal (dengan heuristik admissible)	Boros memori pada graf besar
Cepat untuk ruang pencarian kompleks	Bergantung pada akurasi heuristik
Menggabungkan informasi nyata dan estimasi	Perlu fungsi heuristik yang tepat
Cocok untuk rute, game AI, dan robotika	Tidak efisien jika h(n) buruk

Soal

A. Pemahaman Konsep Heuristik dan Fungsi Evaluasi

1. Jelaskan dengan kata-kata Anda sendiri apa yang dimaksud dengan heuristik dalam konteks kecerdasan buatan, dan mengapa konsep ini penting dalam proses pencarian solusi.

2. Berikan contoh fungsi heuristik (h(n)) dalam kasus nyata (misalnya pencarian rute, permainan, atau navigasi robot), dan jelaskan bagaimana nilai heuristik tersebut membantu mempercepat proses pencarian.
3. Jelaskan perbedaan antara fungsi heuristik (h(n)) dan fungsi evaluasi (f(n)). Sertakan rumus umum yang digunakan serta makna dari setiap komponennya.
4. Heuristik yang admissible dan consistent menjadi syarat penting dalam algoritma A*.

   Jelaskan arti kedua istilah tersebut dan berikan contoh sederhana dari masing-masing.

B. Best-First Search dan A* Algorithm

5. Bandingkan secara singkat antara algoritma Best-First Search dan A* dalam hal:
   - Fungsi evaluasi yang digunakan,
   - Informasi apa yang dipertimbangkan,
   - Apakah hasilnya selalu optimal.
6. Mengapa algoritma **A*** dikatakan lebih efisien dibandingkan Breadth-First Search (BFS) maupun Uniform Cost Search (UCS)? Jelaskan dengan mempertimbangkan peran heuristik.
7. Diketahui graf berikut:

Node	Tetangga (biaya)	h(n)
S	A(2), B(4)	7
A	C(3)	4
B	D(5)	5
C	G(4)	2
D	G(2)	3
G	–	0

Gunakan algoritma A* untuk menentukan jalur terpendek dari S ke G.

Tunjukkan perhitungan nilai g(n), h(n), dan f(n) di setiap langkah.

8. Pada kasus yang sama dengan soal nomor 7, jika digunakan Best-First Search (f(n)=h(n)), jalur mana yang akan dipilih sistem?

   Apakah hasilnya sama dengan A*? Jelaskan perbedaannya.

C. Contoh dan Analisis Pathfinding

9. Dalam kasus pencarian jalur dari satu kota ke kota lain, apa dampaknya jika nilai heuristik terlalu besar (*overestimate*) atau terlalu kecil (*underestimate*) dibandingkan jarak sebenarnya?

   Jelaskan efeknya terhadap kecepatan dan optimalitas hasil pencarian.

10. Sebutkan tiga contoh penerapan nyata algoritma A* dalam kehidupan sehari-hari atau industri, lalu jelaskan bagaimana heuristik diterapkan di masing-masing kasus tersebut.